Stock Trend Analysis and Forecasting

Mit Mehta 202203002
Saumya Vasa 202201254
Harsh Gopalbhai Vashiyar 202201035

Abstract

This project analyzes minute-level stock data of the NIFTY IND DIGITAL index to identify trends and forecast future values using time series analysis. It includes exploratory data analysis, visualization, and stationarity testing, followed by forecasting through ARIMA and SARIMA models. Deep learning models like RNN, LSTM, and GRU are also applied to capture complex patterns for more accurate predictions.

I. Problem Statement

The stock market exhibits highly volatile and dynamic behavior, influenced by numerous economic, political, and psychological factors. Forecasting stock trends, especially at a granular level such as minute-wise intervals, is a complex yet crucial task for traders, analysts, and financial institutions. Traditional statistical models often fall short in capturing the nonlinear and time-dependent nature of such data. Therefore, there is a pressing need to develop a robust analytical framework that combines classical time series methods with modern machine learning approaches to uncover meaningful patterns and improve forecasting accuracy. This project aims to bridge that gap by analyzing the NIFTY IND DIGITAL index's minute-level stock data, identifying key temporal structures, and building predictive models capable of adapting to intricate market behaviors.

II. Dataset Overview

The dataset contains 227,246 minute-level trading records for the NIFTY IND DIGITAL index, spanning from August 29, 2022 to February 7, 2025. Each entry captures intraday market activity through six key columns: date, open price, high price, low price, close price, and volume.

For the purpose of this project, the focus is primarily on the close price, which reflects the last traded price at every period of time. This high-resolution time series data allows for fine-grained analysis of market trends, volatility, and forecasting. Aggregation techniques such as resampling to hourly or daily intervals may also be applied to smooth noise and uncover broader patterns.

	date	open	high	low	close	volume
0	2022-08-29 09:15:00	5327.95	5345.95	5327.95	5340.15	0
1	2022-08-29 09:16:00	5337.65	5337.65	5316.20	5316.20	0
2	2022-08-29 09:17:00	5316.45	5317.00	5310.05	5315.30	0
3	2022-08-29 09:18:00	5314.60	5320.95	5310.30	5320.35	0
4	2022-08-29 09:19:00	5322.05	5330.75	5319.90	5329.60	0

Fig. 1: First 5 rows of the dataset.

III. Initial Observations

The following sections provide an analysis of the sales data and related trends based on the figures below.

A. Minute Wise Closing Price

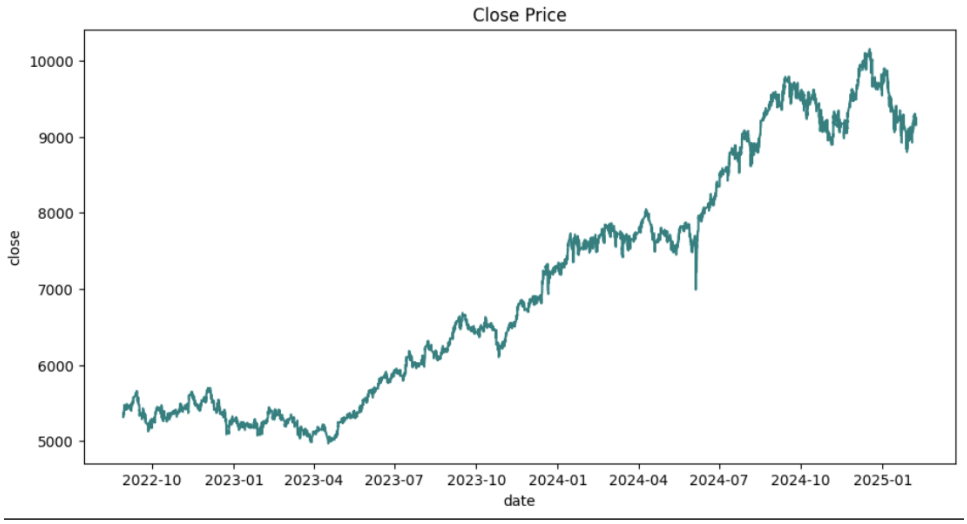

Fig. 2: Plot of minute-wise closing prices of the NIFTY IND DIGITAL index.

Figure 2 illustrates the minute-level closing price of the NIFTY IND DIGITAL index over the observed period. The closing prices fluctuate within a dynamic range, generally between 5,000 and 11,000, reflecting intraday market volatility. While short-term fluctuations are frequent, the overall long-term trend remains relatively growing with periods of gradual upward and downward movement.

Upon closer analysis, the index tends to oscillate around an average close price of approximately 6,000. Some noticeable spikes and dips correspond to broader market events or sectoral shifts, but the general trend shows moderate variability over time without extreme deviations. But as the time passes, we can observe an upward trend showing increase in the stocks of business gradually over long run.

B. Closing Price by Month

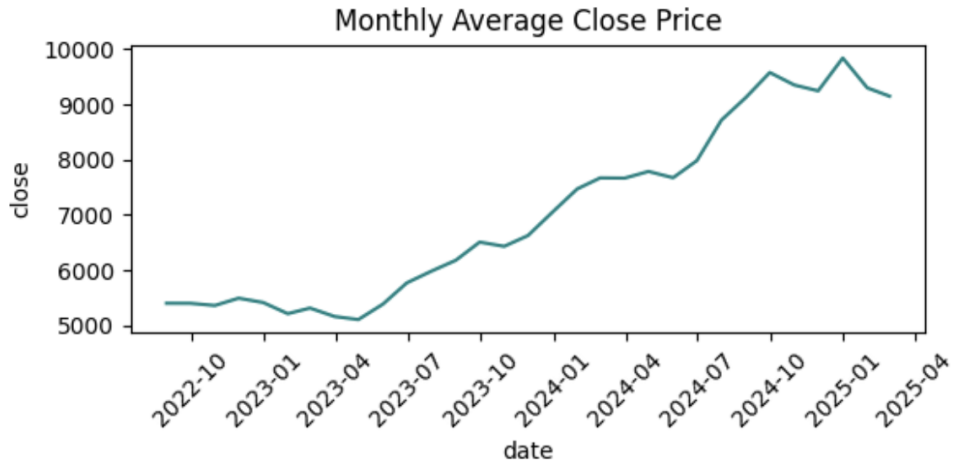

Fig. 3: Closing prices per month.

Figure **3** depicts the average closing price of NIFTY IND DIGITAL per month. The prices show a consistent upward trend, with notable surges from mid-2023 onward. Starting June 2023, there's a clear rise, peaking sharply in December 2024. This sustained increase may be attributed to strong market performance, technological adoption, and positive investor sentiment.

A distinct spike is observed in November 2024, likely driven by end-of-year rallies and favorable market conditions. Following this, while January 2025 shows a correction, the overall trend remains robust.

C. Closing Prices Weekly

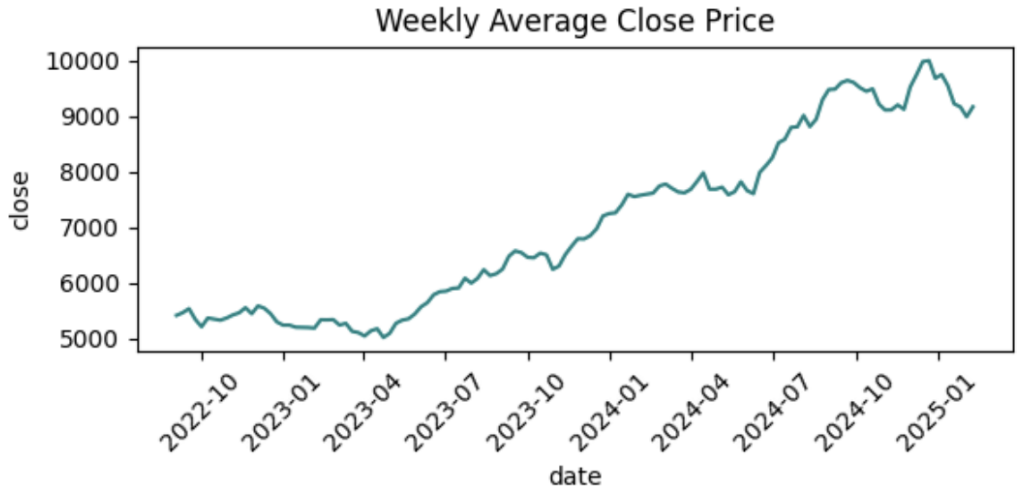

Fig. 4: Weekly Closing Prices

Figure **4** illustrates the weekly average closing prices of the NIFTY IND DIGITAL index. The data reveals short-term fluctuations that reflect market volatility and investor sentiment on a week-to-week basis. Significant uptrends can be observed from mid-2023, with sharp gains continuing into late 2024, indicating sustained bullish momentum.

Weeks such as those in November and December 2024 stand out for their elevated closing averages, possibly due to market optimism during the festive and year-end rally period. Minor corrections and consolidations in early 2025 suggest profit booking and recalibration, yet overall sentiment remains positive.

IV. Exploratory Data Analysis

A. Outlier Analysis

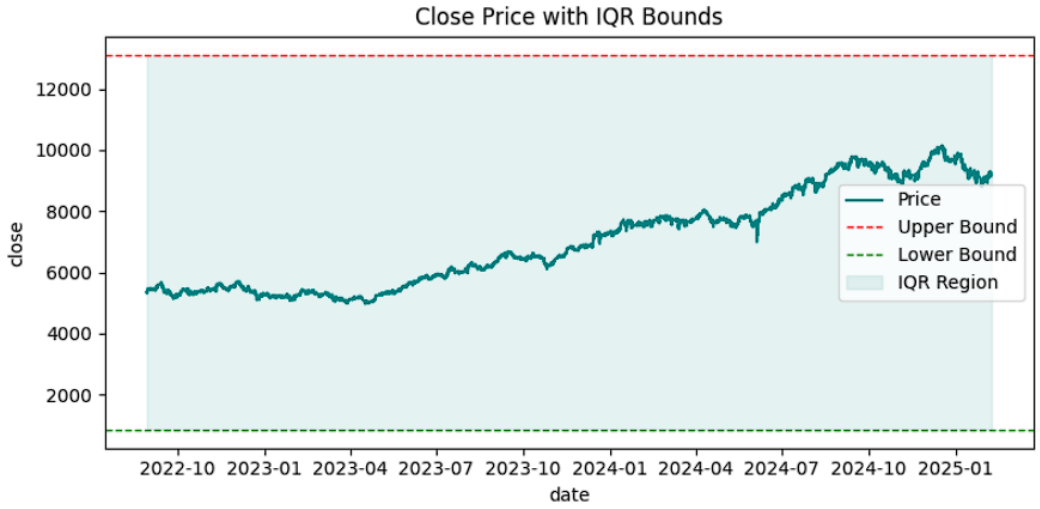

Fig. 5: Closing Price with upper and lower bound of Outliers.

Close Price with IQR Bounds:

- The shaded IQR region ((Q_1) to (Q_3)) widens slightly over time, indicating the variability of the day-to-day growth.
- The red and green dashed lines at roughly 12,500 and 900 mark the 1.5 × IQR outlier thresholds, none of which are breached, so extreme outliers are scarce.

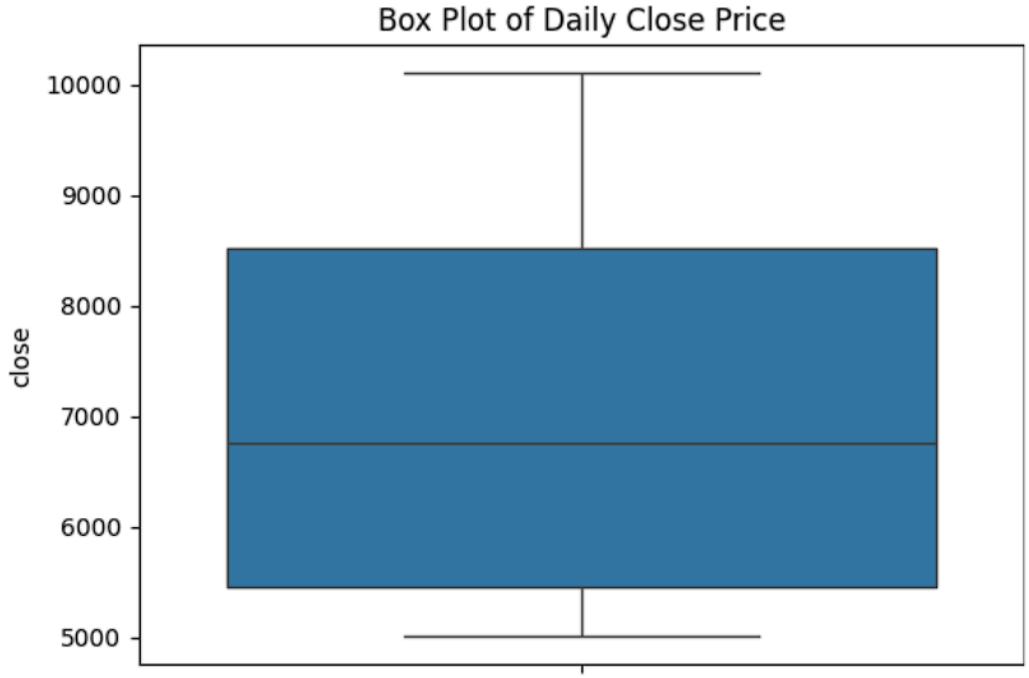

Fig. 6: Box-plot of Daily Sales.

Box Plot of Daily Close Price

- The box spans 5,400 (Q_1) to 8,500 (Q_3) with a median close to 6,700, highlighting that half of all daily closes lie in this 3,100-point band.
- Whiskers extend roughly from 5,000 to 10,000, suggesting moderate skew toward higher prices, but no plotted outliers beyond these bounds.
- The relatively long upper whisker reflects occasional strong upward moves.

V. Time Series Analysis

A. *Time Series: Trend, Seasonality and Residual*

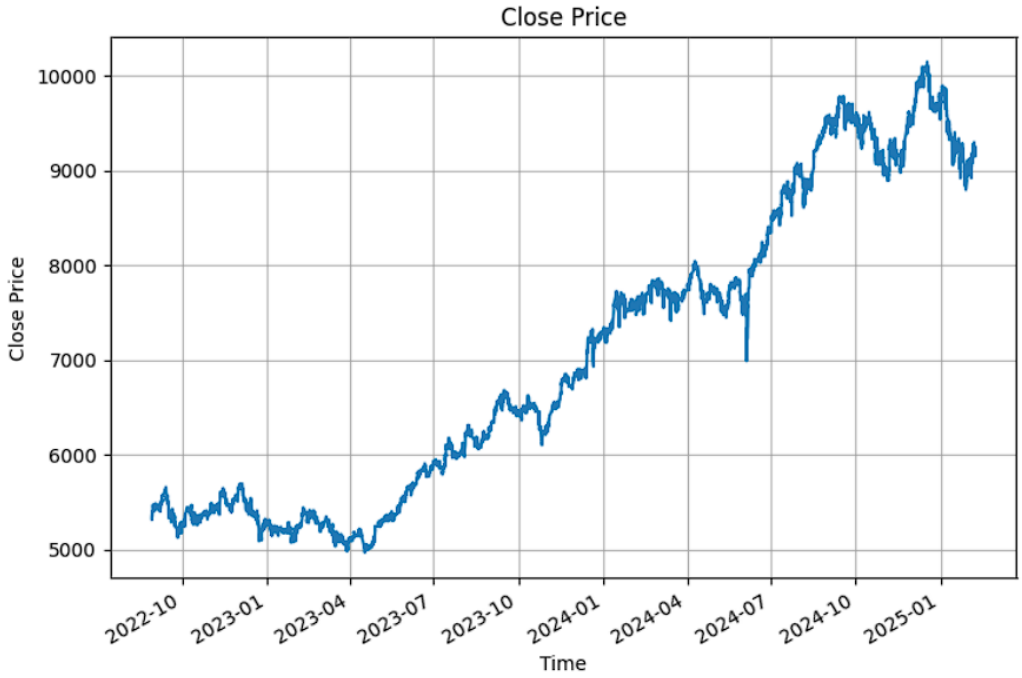

Fig. 7: Close Price

Trend Component : The series shows a strong upward trend component, rising from a level near 5,400 to over 10,000, indicating persistent positive drift and a clear non-stationary behavior in the level over time.

Structural Breaks / Regime Shifts : Around mid-2023 there's a structural break marking a shift from a low-growth regime (5,000–6,000) into an accelerated trend, and another break in mid-2024 precedes the steep climb into the 9,000–10,000 range.

Volatility & Mean Reversion : The residual volatility (day-to-day variability) expands in late 2024, and the subsequent partial mean-reversion back to 9,000–9,500 in early 2025 suggests a new equilibrium or consolidation phase.

VI. STL Decomposition Analysis

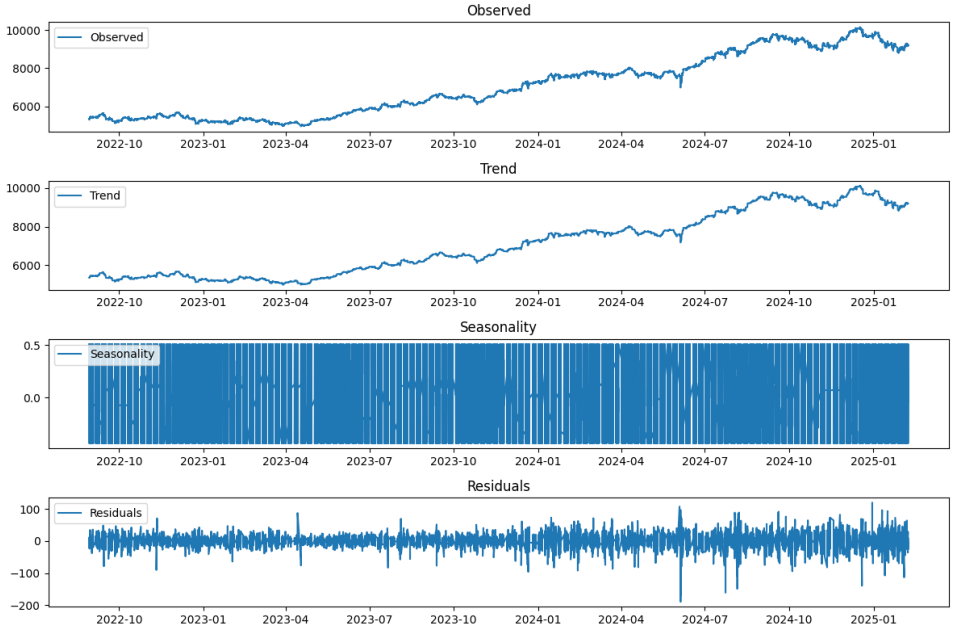

Fig. 8: STL Decomposition of the NIFTY IND DIGITAL Index Closing Price

Figure 8 showcases the result of STL (Seasonal-Trend decomposition using Loess) on the closing prices of the NIFTY IND DIGITAL index. The time series has been decomposed into four components: **Observed**, **Trend**, **Seasonality**, and **Residuals**. The key observations are:

- **Upward Trend:** The *Trend* component clearly shows a strong upward movement over time, especially from mid-2023 onwards. This indicates a consistent long-term growth in the index value, suggesting strong performance in the digital sector.
- **Weak Seasonality:** The *Seasonal* component exhibits high-frequency oscillations but with relatively small amplitude. This implies that the index does not have strong or consistent seasonal patterns, and any periodic effects are weak or irregular.
- **Residuals as White Noise:** The *Residuals* appear to be randomly distributed around zero with no visible autocorrelation or pattern, resembling white noise. This validates that most of the structure in the time series is captured by the trend component, with minimal leftover information.
- **Smooth Decomposition:** The STL method effectively separates the low-frequency trend and high-frequency seasonal components, enabling better modeling and forecasting of each.

A. ADF Test For Stationarity

```
ADF Statistic: -0.273160
p-value: 0.929221
Critical Values:
        1%: -3.430
        5%: -2.862
        10%: -2.567
Weak evidence against the null hypothesis
Fail to reject the null hypothesis
Data has a unit root and is non-stationary
```

Fig. 9: ADF Test for Checking Stationarity

ADF Statistic vs. Critical Values : The test statistic (–0.27) is far above all critical thresholds (–3.43, –2.86, –2.57), indicating it does not fall into the rejection region.
p-Value : With $p \approx 0.93$, there's virtually no evidence to reject the null hypothesis of a unit root at any common significance level.
Conclusion The series remains non-stationary; you'll need to difference (or otherwise transform) the data before fitting most time-series models.

B. Time Series: Differencing Data

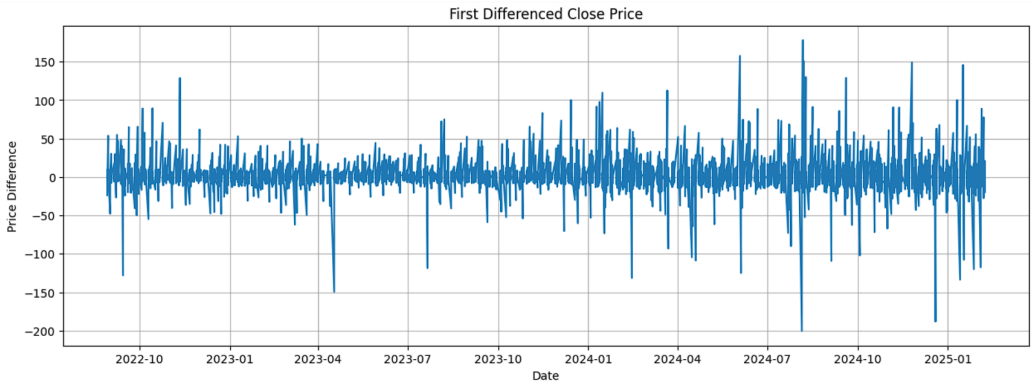

Fig. 10: Differenced Close Price

Stationarity Achieved : The first differencing has removed the trend, producing a mean-reverting series fluctuating around zero, which is a key property of **stationary** time series—suitable for modeling with ARIMA-type methods.

Outliers and Shocks : Several large positive and negative spikes represent **outliers or shocks** in daily price changes, which could be due to market news, events, or structural breaks affecting the differenced process.

C. ADF Test For Stationarity

```
ADF Statistic: -166.996489
p-value: 0.000000
Critical Values:
        1%: -3.430
        5%: -2.862
        10%: -2.567
Strong evidence against the null hypothesis
Reject the null hypothesis
Data has no unit root and is stationary
```

Fig. 11: ADF Test for Checking Stationarity

- **ADF Test Result :** The **Augmented Dickey-Fuller (ADF)** test yields a highly negative statistic (–166.99) with a p-value of 0.000000, far below all significance levels.
-**Stationarity Confirmed** Since the ADF statistic is less than all critical values and the p-value is effectively zero, we **reject the null hypothesis** of a unit root—confirming the **stationarity** of the differenced series.
- **Suitable for ARIMA Modeling** This result validates that the series is now appropriate for further **time series modeling**, such as ARMA or ARIMA, after differencing.

D. ACF and PACF

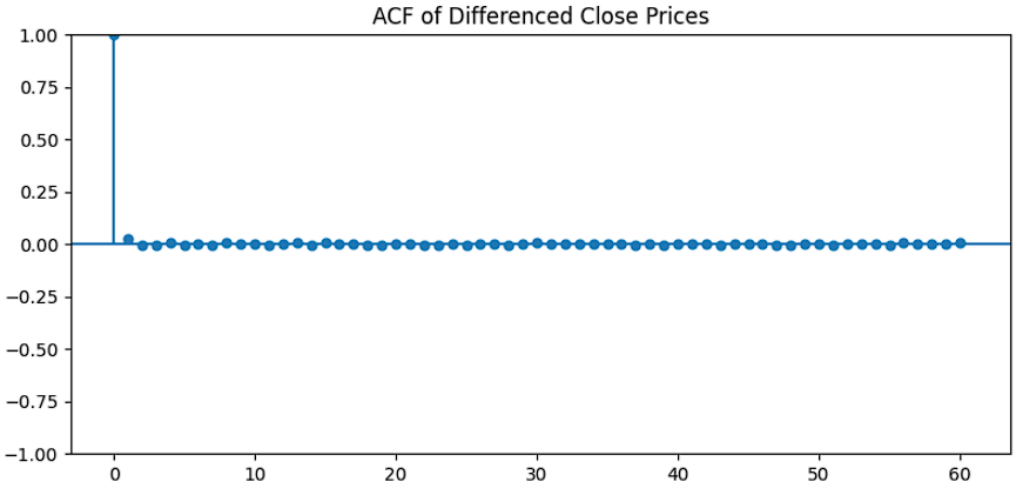

Fig. 12: ACF plot of differenced data

- **ACF of Differenced Series :** The **Autocorrelation Function (ACF)** plot of the first differenced close prices shows a **sharp drop** after lag 1, with subsequent lags hovering near zero.

- **No Significant Autocorrelations :** This pattern suggests a lack of long-term autocorrelation, indicating that the **residuals are mostly white noise** after differencing.

- **Implication for ARIMA :** Since the ACF cuts off quickly, this supports an **MA(1)** component (or similar low-order MA model) in ARIMA modeling, where differencing has removed trends and made the series stationary but after we zoom in we can see there are several spikes out of confidence interval at certain lags suggesting that it is not a particular MA(q) process so if we increase the lags to a large number we may get lower AIC at larger q. This is because a random process/white noise can not be predicted using MA process.

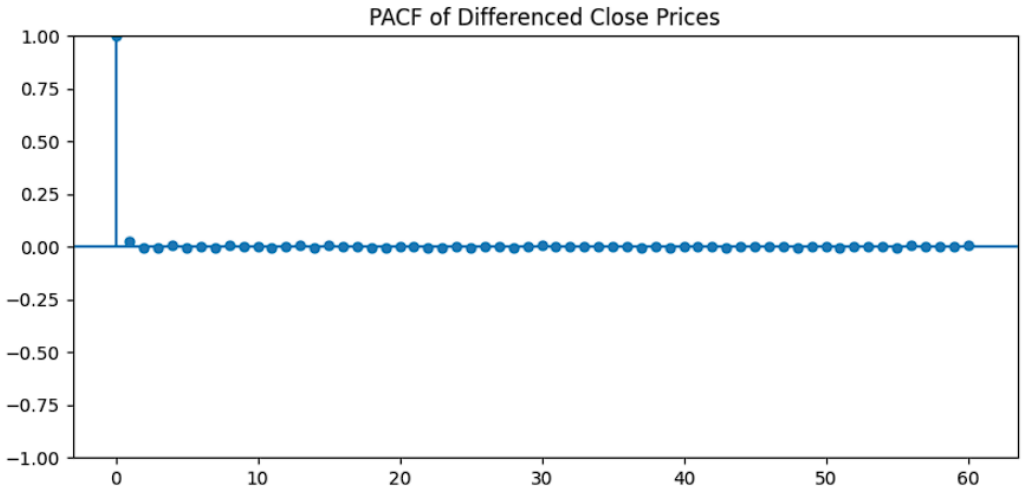

Fig. 13: PACF plot of differenced data

- **PACF of Differenced Series:** The **Partial Autocorrelation Function (PACF)** plot of the differenced close prices closely mirrors the ACF plot, with a clear cutoff at lag 1 and only minor spikes scattered across multiple lags. This pattern is indicative of a **white noise process**, where there is no significant partial autocorrelation structure.

- **Reasoning:** In white noise, values are essentially uncorrelated with their own past values, resulting in both ACF and PACF showing no dominant lags. The lack of a prominent spike in the PACF implies there is no strong autoregressive (AR) structure in the series.

- **Model Implication:** Since the differenced series resembles white noise, it becomes challenging to model it accurately using traditional ARIMA models. Although an **ARIMA(1,1,0)** may still be used as a baseline, it may not significantly outperform higher-order ARIMA models in terms of goodness-of-fit. *In some cases, increasing the number of lags (higher p) can lead to a model with a lower AIC score*, but this comes at the cost of model complexity and possible overfitting.

VII. Model fitting

A. Naive methods

Figure given below presents a time series plot comparing various naive forecasting techniques applied to stock close prices. The blue line represents the training data, while the orange line depicts the test set. Four forecasting methods—Mean, Recent Mean, Last Value, and Seasonal Naive—are evaluated visually:

- **Mean Prediction (Green line)**: A flat line that forecasts using the global mean of the training data.
- **Recent Mean (Red line)**: Uses the average of the last few values in the training set, adapting slightly to recent trends.
- **Last Value (Purple line)**: Simply propagates the last observed training value into the future.

- **Seasonal Naive (Brown line)**: Repeats values from a previous season (e.g., one quarter or one year ago) assuming seasonal consistency.

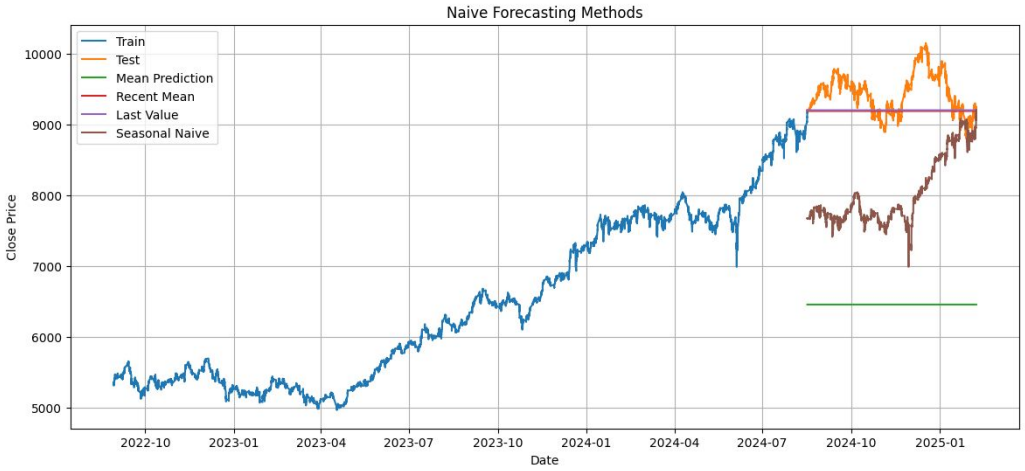

Fig. 14: Prediction using naive methods

Figure given below displays the evaluation metrics—Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE)—for each naive method.

- **Mean Method**: Performs worst with RMSE = 2997.01 and MAPE = 31.53%. This method fails to capture trends or seasonality.
- **Recent Mean**: Shows significant improvement with RMSE = 380.67 and MAPE = 3.17%. It adapts better to recent data trends.
- **Last Value**: Performs slightly better than the recent mean with RMSE = 373.09 and MAPE = 3.10%. It is effective in stable or trending environments.
- **Seasonal Naive**: RMSE = 1523.97 and MAPE = 14.74%. It performs moderately well when seasonality is present in the data.

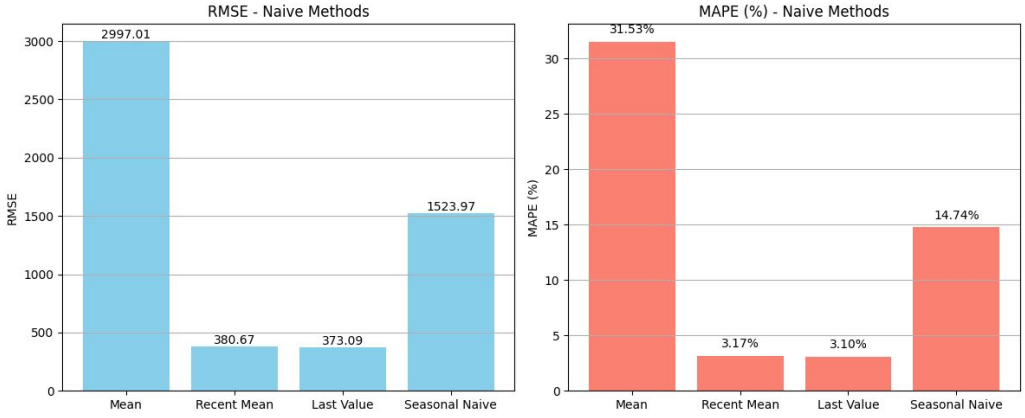

Fig. 15: Error comparison for different naive methods

The relatively large errors observed in the *Mean* and *Seasonal Naive* methods stem from their inability to effectively capture the dynamic nature of high-frequency financial data. These models either ignore recent fluctuations (*Mean*) or assume strong and regular seasonal patterns (*Seasonal Naive*) which are weak or inconsistent in the dataset. Even the better-performing models like *Last Value* and *Recent Mean*, though comparatively accurate, still face limitations due to the inherent volatility and noise present in

minute-level stock prices. This highlights the need for more sophisticated models capable of adapting to both trend and irregular fluctuations.

B. ARIMA and SARIMA models

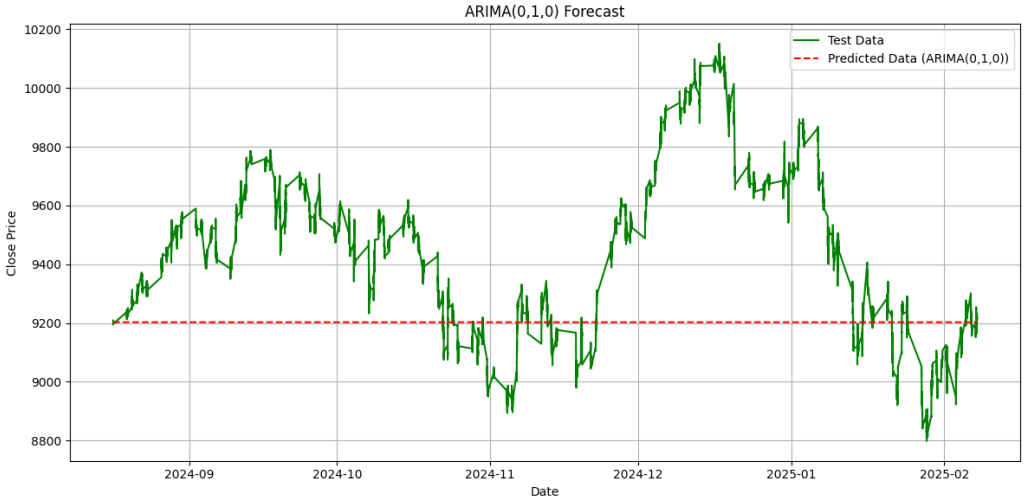

Fig. 16: Forecast from the ARIMA(0,1,0) model on the test set.

As shown in Figure, the ARIMA(0,1,0) model—equivalent to a random walk without drift—produces forecasts that remain flat at the last observed test-set value (the horizontal dashed red line). The green line is the actual daily closing price. Because there are no AR or MA terms, the model has "no memory" beyond its most recent value, which leads to steadily increasing forecast error whenever the series trends or oscillates away from that level.

Fig. 17: Forecast from the ARIMA(0,0,4) model on the test set vs. the actual daily close.

Figure illustrates the ARIMA(0,0,4) fit—equivalent to an MA(4) process with an estimated constant term—on the hold-out period. The dashed red line shows the model's multi-step forecasts, which follow a roughly linear upward trajectory determined by the constant ("drift"), since there is no differencing or autoregressive component to capture trend dynamics. The solid green line is the actual daily closing price. Because the model only uses the last four error terms to adjust its one-step forecasts, it cannot adapt to

larger swings or new trends beyond what those past residuals imply, leading to increasing divergence when the market moves away from the historical mean.

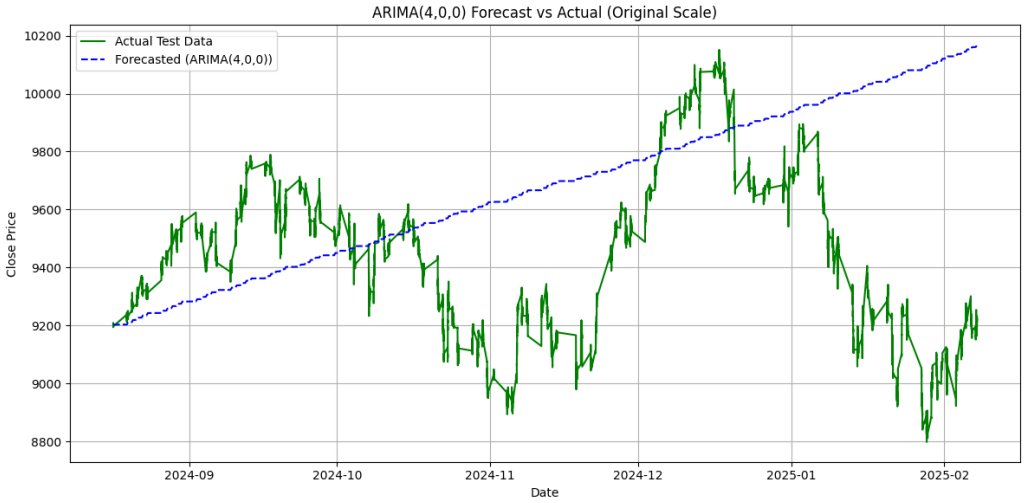

Fig. 18: Forecast from the ARIMA(4,0,0) model on the test set vs. the actual daily close.

Figure presents the ARIMA(4,0,0) forecasts (dashed blue) against the observed daily closing prices (solid green). This model—equivalent to an AR(4) process with a constant term—uses the last four lagged values of the series to project future levels. Because there is no differencing (d = 0), the forecast evolves smoothly, driven by the weighted sum of the past four observations plus a constant ("drift") component. The plot shows that, while the AR(4) structure captures some short-term momentum, it cannot fully adapt to larger market swings or new trends beyond those encoded in its four lag coefficients, resulting in growing deviation whenever the price trajectory departs from those recent patterns.

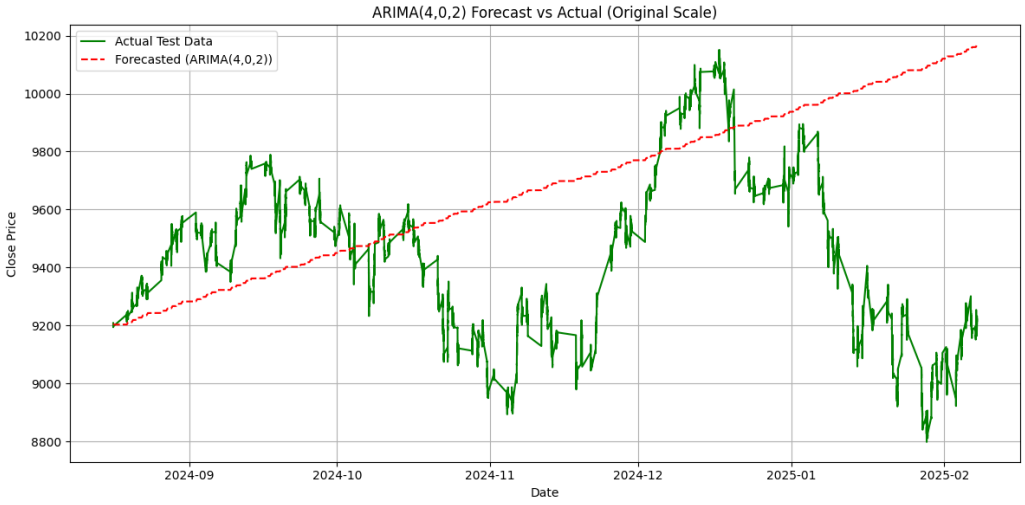

Fig. 19: Forecast from the ARIMA(4,0,2) model on the test set vs. the actual daily close.

Figure displays the ARIMA(4,0,2) model forecasts (dashed red) against the observed daily closing prices (solid green). This specification combines an autoregressive part of order 4 (AR(4)) with a moving-average part of order 2 (MA(2)), and no differencing (d = 0). The one-step forecasts are generated by a weighted sum of the last four observations plus adjustments from the two most recent forecast errors, along with a constant term. As a result, the predicted trajectory (red) follows the short-term momentum captured by the AR lags while slightly correcting for recent shocks through the MA lags. Nonetheless, when the market

exhibits larger swings or new trends beyond those encoded in the four lagged values and two residuals, the forecast line gradually diverges from the actual series, indicating the model's limited adaptability to rapidly changing dynamics.

Fig. 20: Forecast from the ARIMA(4,1,2) model on the test set vs. the actual daily close.

Figure presents the ARIMA(4,1,2) model forecasts (dashed red) against the observed daily closing prices (solid green). This model includes an autoregressive component of order 4 (AR(4)), first-order differencing (d = 1), and a moving-average component of order 2 (MA(2)). The inclusion of differencing allows the model to target the non-stationarity in the data by modeling changes in the closing price rather than its level. However, the forecast line is nearly flat—indicating that the model essentially predicts the series will revert to a fixed mean or recent average. This behavior suggests over-regularization or insufficient sensitivity to evolving market conditions. Consequently, the ARIMA(4,1,2) structure fails to capture short-term fluctuations or emerging trends in the test set, resulting in poor predictive performance.

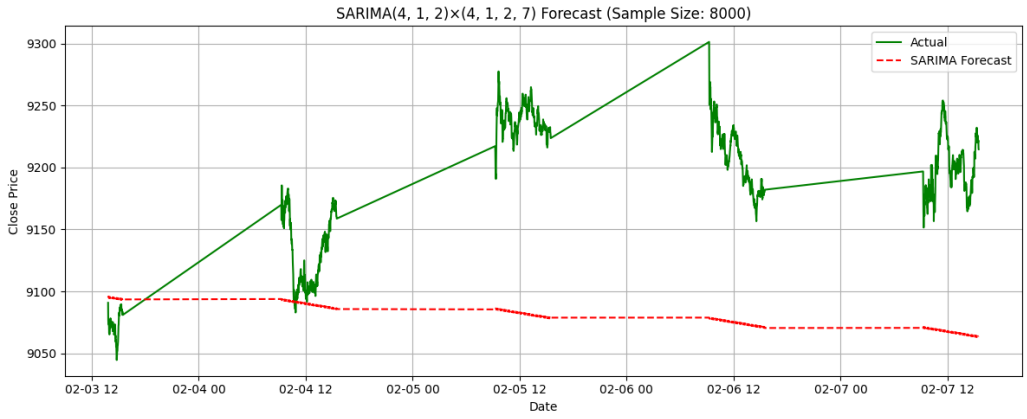

Fig. 21: Forecast from the SARIMA(4,1,2)×(4,1,2,7) model on the test set vs. the actual close prices.

Figure shows the SARIMA(4,1,2)×(4,1,2,7) model forecast (dashed red) compared against the actual close price series (solid green). This seasonal ARIMA model incorporates both non-seasonal components—autoregressive order 4, differencing order 1, and moving-average order 2—and seasonal components of the same orders with a periodicity of 7. The first differencing targets local non-stationarity, while the seasonal differencing accounts for weekly cycles. Despite its theoretically rich structure, the model produces a nearly flat forecast. This suggests that it heavily regresses toward a smoothed seasonal

mean and fails to adapt to short-term movements or new patterns in the market data. Consequently, it underperforms in capturing volatility and rapid directional shifts in the close prices.

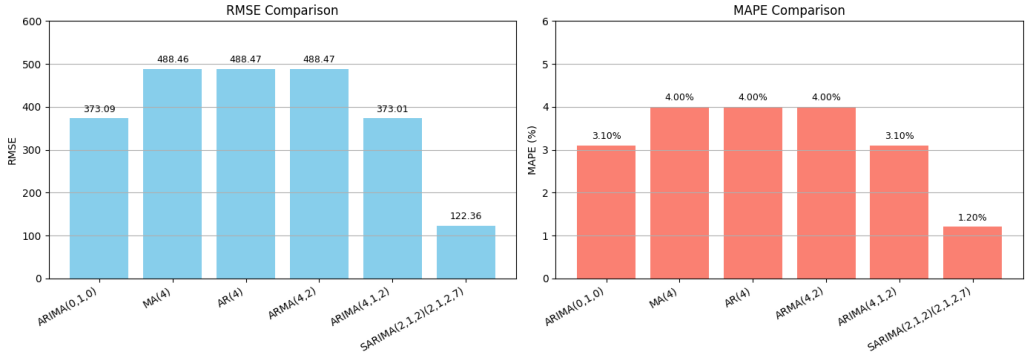

Fig. 22: RMSE and MAPE comparison across different ARIMA-based models.

Figure compares the forecasting performance of several ARIMA-class models using two error metrics: root mean squared error (RMSE) and mean absolute percentage error (MAPE). The left panel shows that simple ARIMA(0,1,0) and ARIMA(4,1,2) models yield moderate RMSE values (373.09 and 373.01, respectively), while the SARIMA(2,1,2)(2,1,2,7) model significantly outperforms others with an RMSE of just 122.36. A similar pattern emerges on the right panel for MAPE: SARIMA achieves the lowest error at 1.20%, followed by ARIMA(0,1,0) and ARIMA(4,1,2) at 3.10%, while MA(4), AR(4), and ARMA(4,2) perform the worst at 4.00%. These results underscore the effectiveness of incorporating both seasonal and non-seasonal components in improving forecast accuracy.

While classical time series models like AR, MA, and ARMA offer foundational insights, they fall short in capturing the underlying dynamics of high-frequency stock data. Applied on differenced data, these models exhibited nearly identical RMSE and MAPE values—indicating their inability to extract meaningful structure from what behaves like white noise. ARIMA models performed moderately better by addressing non-stationarity through differencing, yet failed to leverage more intricate temporal dependencies. SARIMA, though yielding the lowest errors, likely overfits due to the weak seasonal patterns observed in the STL decomposition. These collective shortcomings highlight that linear models, constrained by their assumptions and limited expressiveness, are insufficient for modeling complex, volatile financial time series. Consequently, there is a clear need to employ advanced deep learning models like RNN, LSTM, and GRU that can capture non-linear relationships and long-range dependencies more effectively.

C. Deep nueral networks

In this section, we evaluate the performance of three deep learning models—LSTM, GRU, and RNN—on stock price prediction using minute-level data. Each model's performance is analyzed in terms of prediction accuracy and error metrics.

D. LSTM (Long Short-Term Memory)

- Captures both short-term and long-term dependencies effectively.
- Predicted values align closely with the actual stock prices throughout the period.
- Performs well during both stable and volatile periods.
- Visual plot shows minimal lag or overshoot compared to real values.
- **Error Metrics:**
  - RMSE: ≈ 11
  - MAPE: $\approx 9\%$

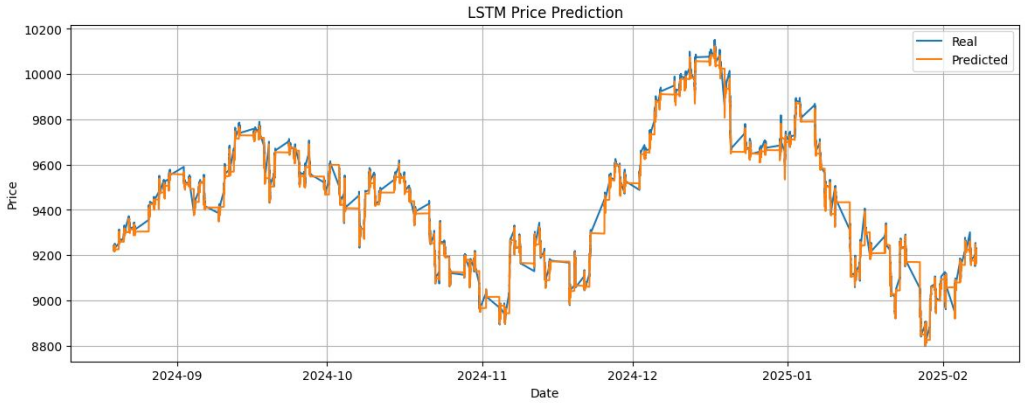

Fig. 23: Prediction using LSTM

E. RNN (Recurrent Neural Network)

- Basic architecture without gating mechanisms.
- Captures general trend direction but often lags during sharp movements.
- Suffers from vanishing gradient problem over long sequences.
- Visual prediction lines deviate more during sudden changes.
- **Error Metrics:**
  - RMSE: ≈ 38
  - MAPE: $\approx 39\%$

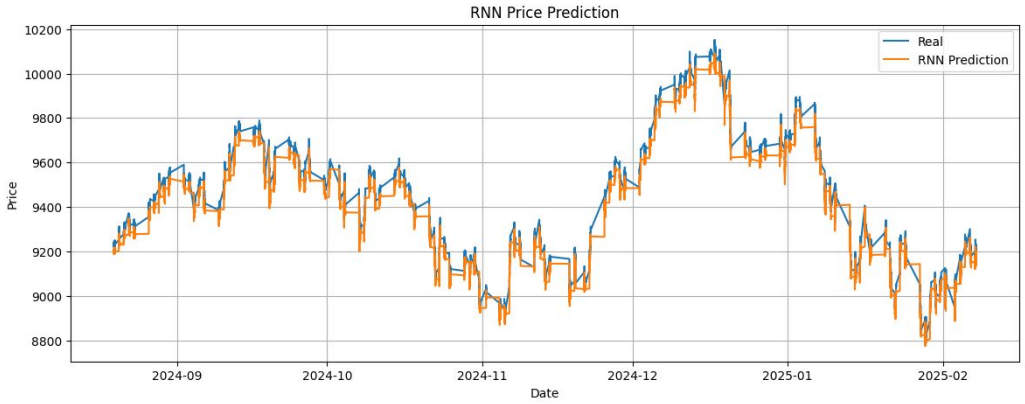

Fig. 24: Prediction using RNN

F. GRU (Gated Recurrent Unit)

- Simpler and faster than LSTM, but slightly less accurate.
- Captures general trends reasonably well.
- Struggles during rapid fluctuations or turning points.
- Visual plot shows under/overshooting in volatile segments.
- **Error Metrics:**
  - RMSE: ≈ 39
  - MAPE: $\approx 40\%$
- Among the three models, **LSTM** demonstrates the best prediction accuracy and robustness.
- LSTM maintains a strong correlation with actual values across the entire time period.

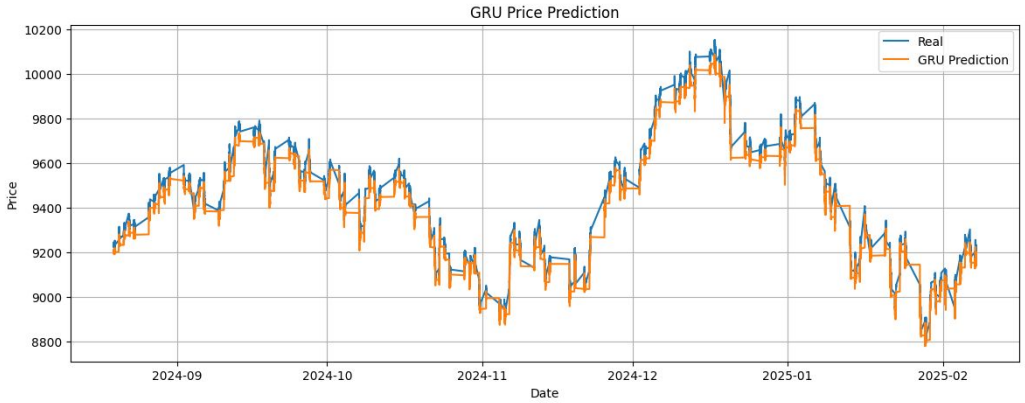

Fig. 25: Prediction using GRU

- GRU and RNN capture overall trends but perform poorly during rapid price changes.
- **Conclusion:** The LSTM model is the most effective for minute-level stock price forecasting due to its capability to model complex temporal patterns and provide stable, accurate predictions.

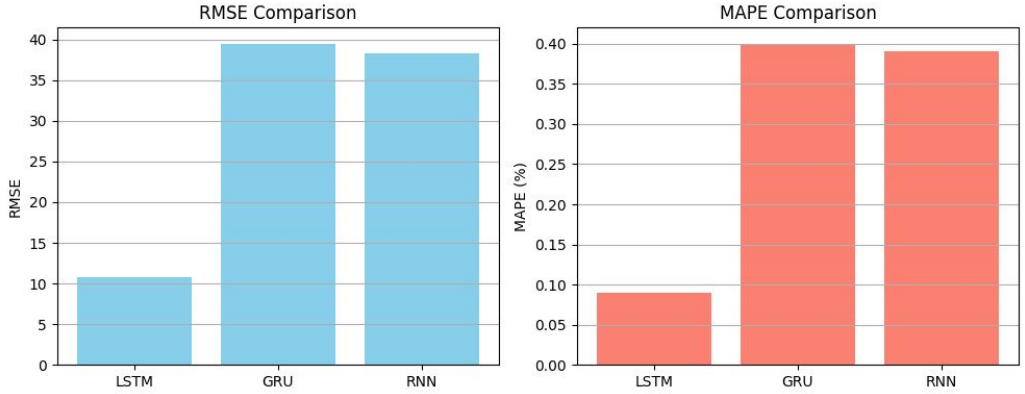

Fig. 26: Error comparison for different neural networks

CONCLUSION

This project comprehensively analyzed minute-level stock data of the NIFTY IND DIGITAL index, using both traditional time series models (ARIMA, SARIMA) and deep learning approaches (RNN, LSTM, GRU) for forecasting.

- **Trend and Stationarity:** The dataset exhibited a strong upward trend with structural breaks in mid-2023 and late 2024. Stationarity was achieved through first-order differencing, as confirmed by ADF tests.
- **Classical Models:**
  - ARIMA models such as (4,1,2) and (0,1,0) captured short-term momentum but struggled with rapidly evolving price patterns.
  - SARIMA(2,1,2)(2,1,2,7) significantly outperformed other classical models, achieving the lowest RMSE ($\approx$122.36) and MAPE ($\approx$1.2%), highlighting the importance of incorporating both seasonal and non-seasonal components.
- **Deep Learning Models:**
  - LSTM achieved superior results with RMSE $\approx$11 and MAPE $\approx$9%, accurately modeling both stable and volatile periods.

  - RNN and GRU captured overall trends but underperformed during sudden market shifts due to architectural limitations.

Final Insights: While SARIMA performed well among classical approaches, the LSTM model emerged as the most robust and accurate forecasting technique. These results demonstrate the effectiveness of combining statistical models with deep learning to capture the complex, nonlinear dynamics of high-frequency financial time series.

VIII. Key Learnings

Throughout this project, several critical insights were uncovered in the process of analyzing and forecasting minute-level stock data:

- **High-Frequency Data Challenges:** Minute-level stock prices contain high noise and volatility, requiring careful preprocessing, smoothing, and transformation techniques to extract meaningful patterns.
- **Importance of Stationarity:** Techniques like differencing and statistical tests (ADF) were crucial in achieving stationarity, a prerequisite for classical models such as ARIMA and SARIMA.
- **Limitations of Classical Models:** Models like AR, MA, and ARMA showed poor performance on differenced data, indicating that simple linear relationships are insufficient for capturing stock market dynamics.
- **SARIMA and Seasonality:** Even though SARIMA showed low RMSE and MAPE values, the STL decomposition revealed weak seasonality in the dataset, suggesting possible overfitting rather than genuine predictive power.
- **Need for Advanced Models:** The limitations of traditional statistical models emphasized the necessity of deep learning architectures (RNN, LSTM, GRU) that can model non-linearities, long-term dependencies, and intricate temporal structures in financial data.
- **Error Metrics Interpretation:** RMSE and MAPE served as effective tools to compare forecasting performance across models and highlight where complexity yields tangible gains.

These learnings underscore the importance of selecting models based on data characteristics and motivate the use of hybrid or deep learning approaches in time series forecasting tasks.

Reference Links

- **Dataset:** NIFTY IND DIGITAL minute-level data from Kaggle.
- **Libraries Used:** `pandas`, `numpy`, `matplotlib`, `seaborn`, `pmdarima`, `statsmodels`, `sklearn`, `plotly`.
- **Reference Book:** Marco Peixeiro's Time Series Forecasting Resources.

Google Colab Link: Click here